Tutorial & Documentation

WSAG - Multi Player Online Network

WSAG 4.x

by

OLDJIM

www.oldjim.ch/WSAG

INDEX

About WSAG....3
How to setup your Router for WSAG to work....3
WSAG Router settings:....4
Player Setup....5
The players Elements:....5
Controller:....5
Player:....5
Camera:....5
HUD:....5
Shoot:....5
ShootClient:....6
SyncTime:....6
The Players LogicBricks....7
Controller:....7
Player:....8
Camera:....9
HUD:....9
Shoot:....9
ShootClient:....11
SyncTime:....11
The players in Action:....12
Spawn as Explorer or Your Controlled Client :....12
Spawn as external Client:....12
The 1-ModeSetup.py Script....12
def SetLocation():....12
def AddMode():....12
def SetPlayer():....12
The 1-SetMenuProp.py Script....12
The 2-MouseMove.py Script....12
def OnOff():....12
def MouseY():....12
def MouseX():....12
The 2-Move.py Script....13
The 2-OriPos.py Script....13
def Main():....13
def YourOriPosShoot():....13
def SetOriPosShoot():....13
def YourOriPos():....13
def SetOriPos():....13
The 3-WSAG.py Script....13
def Main():....13
def IPLAN():....13
def Shoot():....13
def IPWeb():....13
def Port():....13
def Receive():....13
def Send():....14
def SetIP():....14
def AppendIP():....14
def ConnetMe():....14
def YourConnection():....14

def SetData(): 14
def SyncTime(): 14
def Disconnect(): 14
def ServerConnect(): 14
def ClientConnect(): 14
def ServerSet(): 14
def ClientSet(): 14

About WSAG

The goal of WSAG is to make it possible for everyone to make a A Solid Multi Player Online Network Game. This Document tries to give everybody the Information that is needed to work whit the WSAG Setup. It aims at beginners and pros. It contains:

How to Use:
A section that will give you a simple Step by Step guide on how to integrate WSAG in to your own Game.

How it Works:
This section will explain every Logic Brick and Line of Code and how the work together inside of the WSAG setup.

Hope you enjoy this documentation and It helps you to get started whit WSAG.
If any Questions come up feel free to ask over at:
http://www.gameblender.org/viewtopic.php?f=22&t=906
And if you have an suggestions how to improve this Documentation or the WSAG setup itself please post it at:
http://www.gameblender.org/viewtopic.php?f=22&t=626

How to setup your Router for WSAG to work

Your firewall is like a row of ninja assassins just waiting to kill unauthorized messengers and spies. I will try to visualize this using smilies:

You may now ask your self why there are 2 rows of ninjas
A Network that runs over a Router has one Firewall on the Computer and a Second one on the Router. Between these these 2 firewalls is the hardworking router penguin of absolute control, like so:

=Router firewall

= penguin of absolute control

=Computer firewall

To open a port of your computer fire wall you need to go in to the control-panel of your computer and find the fire wall settings.
There you add the port or port-range that you like to open up and assign the Titel to it and what Protocol it uses (TCP or UDP (or both))
Afther you have done this your firewall setup will look like this:

=Router firewall

= penguin of absolute control

=Computer firewall

Note an the side:
There: some Computer are setup so that you don't have to deal with its Firewall or the firewall is even deactivated. If you have a computer like that, the setting above will not be needed - but your computer will be at a bigger risk on your everyday Internet use. So you should consider to setup a firewall that you have the control over, to optimize your security not because of WSAG but because of your normal Internet consumption.

Whit this said lets move on and configure the router firewall:

To login to your router your open a Web-browser and enter the Routers IP in to the address field. Usually thats 192.168.1.1.
Now you will need to enter the routers Login name and Password (you will find these informations in the user-manual of your Router)
Now find the menu where you can forward port and enter the same information you entered for the computer-firewall. Whit the addition that you will have to define the Computers IP. This is because a router can have more then one PC connected to it.
Now your Router looks like this:

= router firewall

=penguin of absolute control

=computer firewall

Now any message that comes over these ports will be forwarded by our penguin of absolute control.

Like this:

= router firewall

=penguin of absolute control

=computer firewall

= You like to send a balloon to a friend.
The Port you opened lets the balloon trough to the penguin of absolute control.
The penguin of absolute control pases it trough the Port you opened in the Router.
The balloon is on its way.
The same process works if a friend sends a balloon to you.
Note this balloon needs tho have the exact information of Its destination and how to get there, Only if the Information of Port and IP are correct it the penguin of absolute control will pass it on.
the penguin of absolute control will never let a balloon that has port 105 in its information trough port 106, even if Port 105 is closed and 106 is open. All the Information has to match.

Thanx nijineko for the base of this tutorial that you made available at:

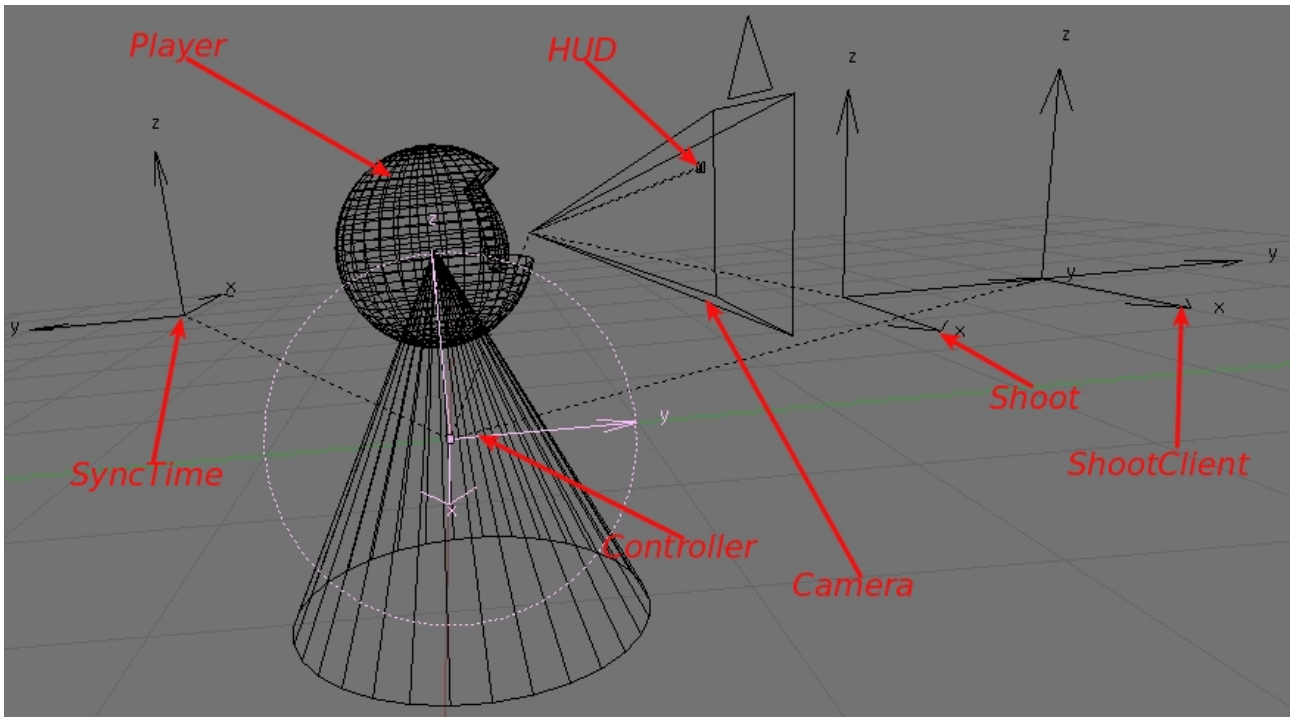
http://www.gameblender.org/viewtopic.php?f=22&t=7#p530

WSAG Router settings:

Port range: 5000 -7000

Protocol: UDP

Player Setup



The players Elements:

Controller:

Keyinputs that effect the Player run over this element.
This Element contains the physics of the Player.
The Bounding Box of this Element is set to the size of the Player

Player:

Is a Child of the Controller.
Its property's are the Data that will be send over the network.

Camera:

Is a Child of the Controller.
It gives you the First Person View.
Mouse-movement controls its View-angle.
The Mouse-move-control can be turned on/off whit the Tab-Key

HUD:

Is a Child of the Camera.
Shows the Game-Point count.

Shoot:

Is a Child of the Camera.
When triggered it shoots the Object Bullet in the direction that its pointed.
The Object Bullet is on a different Layer.

ShootClient:

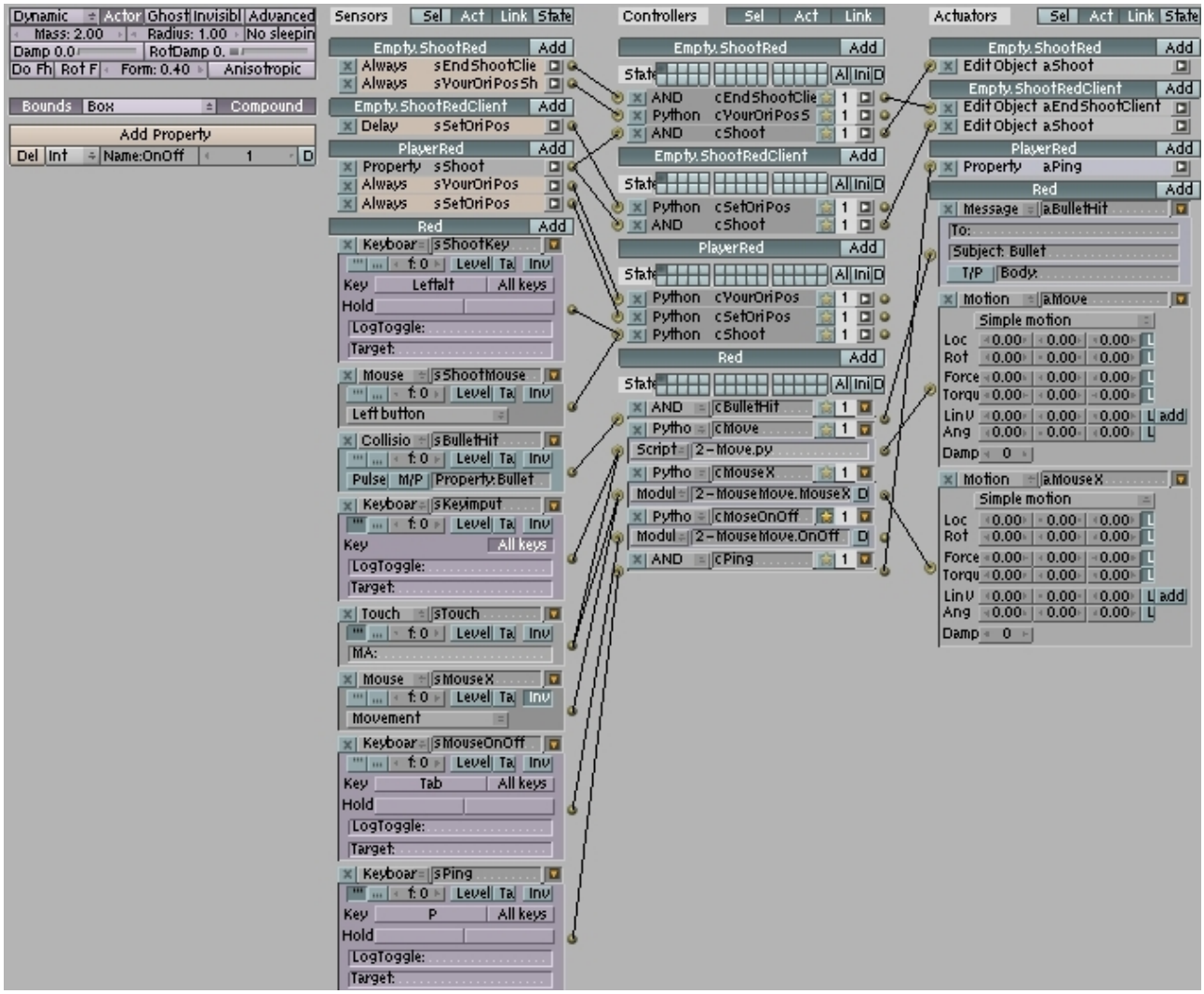
Is a Child of the Player.
Same as Object Shoot but it will be destroyed if Object Shoot is also Created.
It only remains if the Player is created as a client (not the one your controlling)

SyncTime:

Is a Child of the Player.
In Client Mode it sends a Ping Signal if the P-Key is pressed.
This Signal is used to calculate the SyncTime.
The SyncTime is the time it takes to send data to the server and receive it back.

The Players LogicBricks

Controller:



The Sensors sShootKey and sShootMouse are triggered if you Press the left Alt-Key or the LMB.
They activate the Controller cShoot that is a LogicBrick of the Player,
The Controller cShoot runs the Python Module 3-WSAG.Shoot

The Sensor sBulletHit is triggered if you collide with a Object that has the property Bullet.
Via the controller cBulletHit it activates the Actuator aBulletHit which Sends a Message with the

Subject Bullet. This Message is received by the HUD Sensor sBulletHit.

Any Key will Trigger the Sensor sKeyinput.
The Sensor sTouch is positive if the the bounding box of this Object is touching a other Object.
Both Senors will activate the Python Scrit 2-Move.py
The Controller cMove that runs the Python Script 2-Move.py is connected to the Actuator aMove.

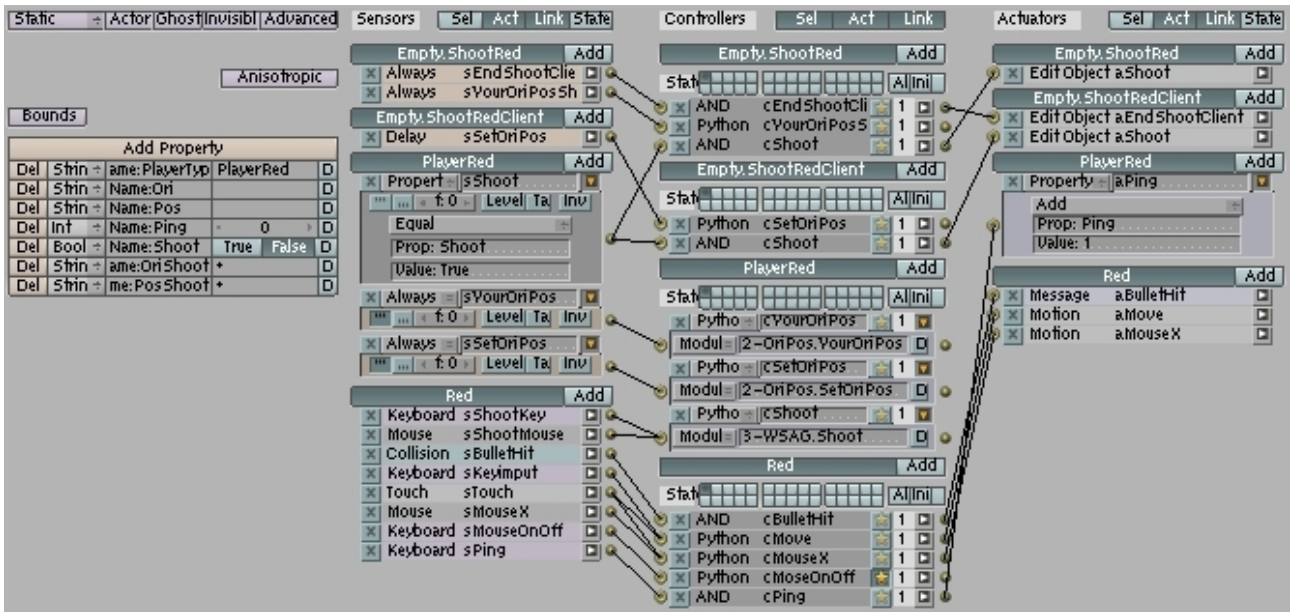
The sMouseX Sensor reacts if the Mouse is moved.
The Sensor sTouch is positive if the the bounding box of this Object is touching a other Object.
Both Senors will activate the Python Module 2-MouseMove.MouseX
The Controller cMouseX that runs the Python Script 2-MouseMove.MouseX is connected to the Actuator aMouseX.

The sMouseOnOff Sensor is triggered if the Tab-Key is pressed and will activate the Controller cMouseOnOff that will run the Python Module 2-MouseMove.OnOff

The sPing Senor is triggered if the P-Key is pressed. This Sensor will activate the Controller cPing. The cPing Controller will activate the Actuator aPing that is a LogicBrick of the Player. This actuator will then add the Value 1 to the property Ping.

The Property OnOff will be used in the Python Module 2-MouseMove.OnOff

Player:



The Sensor cShoot is Triggered if the Property Shoot is True.
If this sensor is triggered it will activate the cShoot Controller of Shoot and ShootClient.
These Controllers then make the Actuator aShoot add a Bullet. (This will ably to Shoot or ShootClient for only one of them will be active at a time but this will be shown later see here)

By the way the Bullet that will be added is on a different Layer the the Players Elements and its Name is defined in the LogicBrick Actuator aShoot.

The Always Sensor YourOriPos will trigger the Python Module 2-OriPos.YourOriPos

The Always Sensor SetOriPos will trigger the Python Module 2-OriPos.SetOriPos

The Players Property's:

These Propertys are used to set your external-clients so that they stay in Sync whit your Controlled Client more about this can be found in the Chapter What Information will be send over the

Network:

The Propery PlayerTyp has to be equal to the name of the Player. For it is used to define what should be creaded on the server and the other client Computers (more information about this can be found in the Chapter Spawn as external Client)

The Property Ori This value is set by the Python Module 2-OriPos.YourOriPos and will be used to set the external-clients Orientation with the Python Module 2-OriPos.SetOriPos

The Property Pos This value is set by the Python Module 2-OriPos.YourOriPos and will be used to set the external-clients Position with the Python Module 2-OriPos.SetOriPos

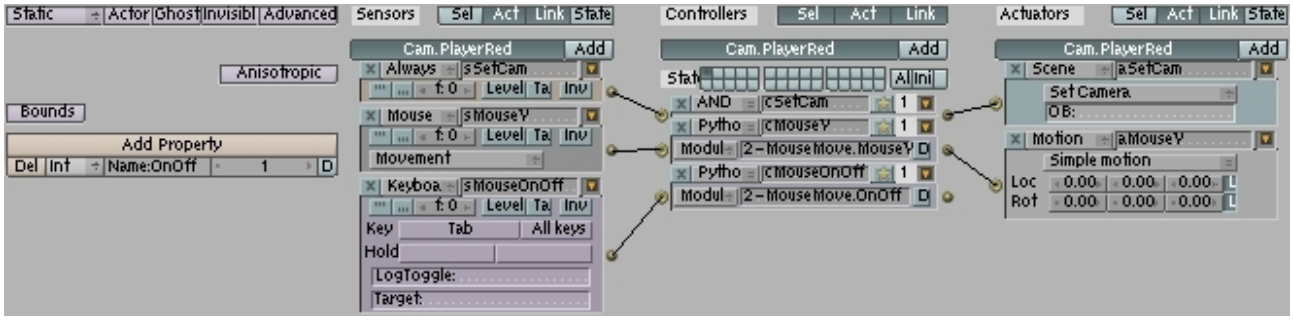
The Property Ping its value is set the Actuator aPing. This Actuator is triggered by controller cPing and Sensors sPing of the controller element of this player setup. The SyncTime element then uses this property to trigger the calculation of your Synctime using the Python Module 3-WSAG.SyncTime

The Property Shoot is set to True or False in the Python Module 3-WSAG.Shoot. and will trigger the Sensor sShoot if true.

The Property OriShoot This Value is set by the Python Module 2-OriPos.YourOriPosShoot and will be used to set Orientation the ShootClient on your external-clients using the Python Module 2-OriPos.SetOriPosShoot

The Property PosShoot This Value is set by the Python Module 2-OriPos.YourOriPosShoot and will be used to set Position the ShootClient on your external-clients using the Python Module 2-OriPos.SetOriPosShoot

Camera:



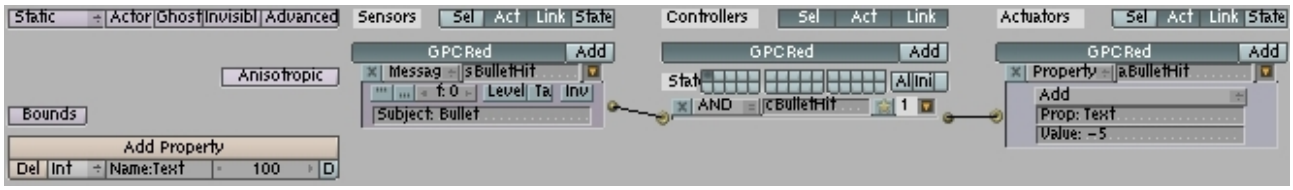
The Sensor sSetCam is triggered onetime at Startup. It will activate the Controller cSetCam.
The Controller cSetCam will run the Actuator aSetCam.
The aSetCam Actuator will Set the Camera to be the Active Camera.

The Sensor sMouseY is Triggered by any Mouse movement.
If Triggered it will Activate the Python Module 2-MouseMove.MouseY which is attached to the Actuator aMouseY.

The Sensor sMouseOnOff is triggered if you press the Tab-Key.
If Triggered it will run the Python Module 2-MouseMove.OnOff

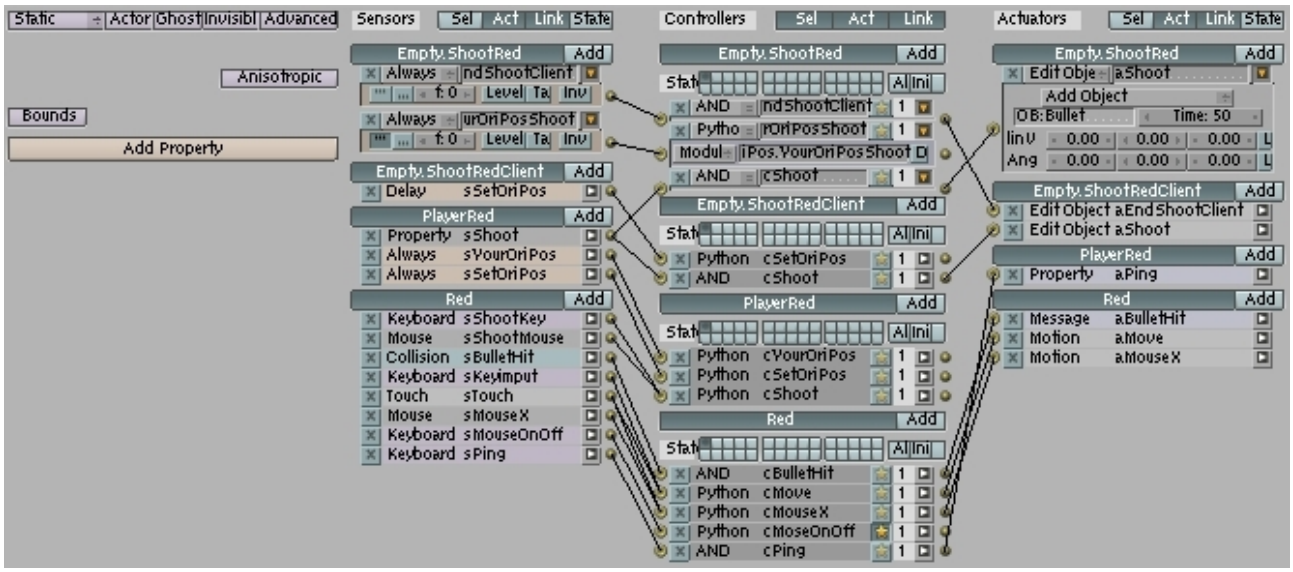
The Property OnOff will be used in the Python Module 2-MouseMove.OnOff

HUD:



The Sensor sBullet hit is triggered if it receives a message with the subject Bullet.
If Triggered it will activate the Controller cBulletHit.
The Controller cBulletHit will then activate the Actuator aBulletHit.
The Actuator aBulletHit will change the Value of the Property Text by -5.

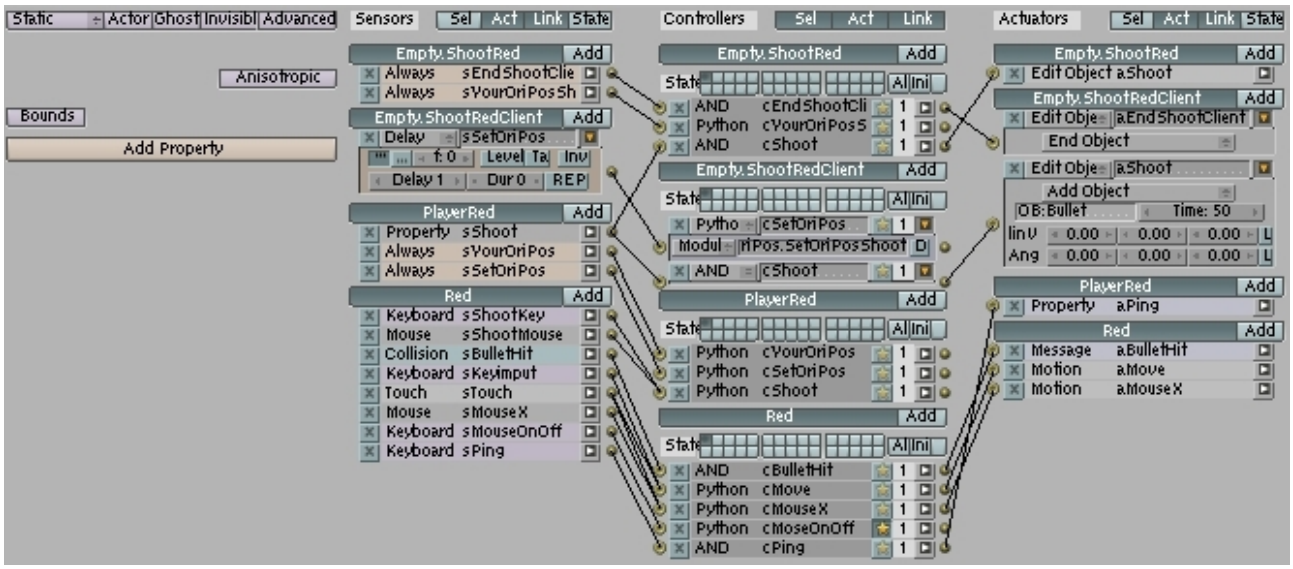
Shoot:



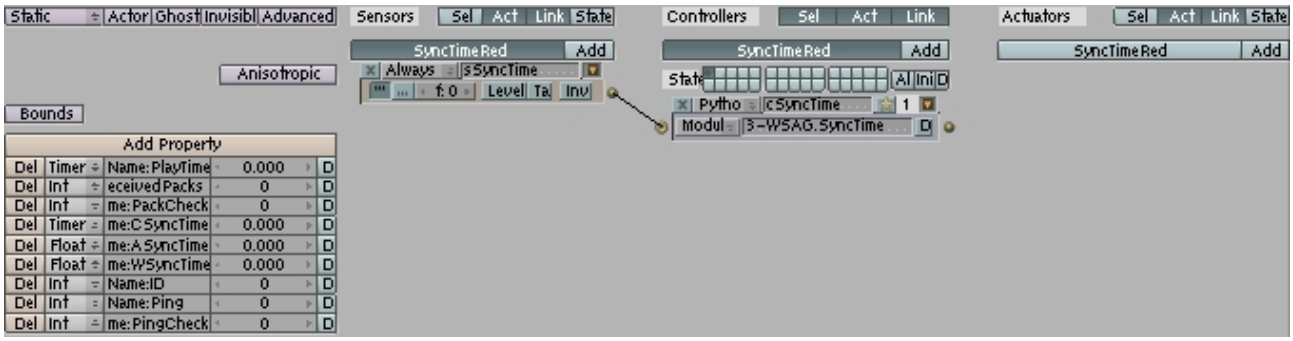
The Sensor sEndShootClient will run onetime at start up and trigger the Controller cEndShootClient. The Controller will then activate the Actuator aEndShootClient which is a LogicBrick from ShootClient. This Actuator will End The Object ShootClient.
More Infos about why and wen the Object ShootClient is ended can be found in the Chapter: Spawn as Explorer or Your Controlled Client

The Controller cYourOriPosShoot is triggered every tick-rate by the Always Sensor sYourOriPosShoot and will run the Python Module 2-OriPos.YourOriPosShoot

ShootClient:



After a delay of one tick-rate the sensor sSetOriPos will trigger the Controller cSetOriPos every tick-rate. This controller will run the Python Module 2-OriPos.SetOriPosShoot:



SyncTime:

The Always Sensor sSyncTime triggers the Controller cSyncTime every tick-rate.

The Controller cSyncTime will run the Python Module 3-WSAG.SyncTime

The Property PlayTime is used in the Python Module 3-WSAG.SyncTime

The Property ReceivedPacks is used in the Python Module 3-WSAG.ClientSet and 3-WSAG.SyncTime

The Property PackCheck is used in the Python Module 3-WSAG.SyncTime

The Property CsyncTime is used in the Python Module 3-WSAG.SyncTime

The Property AsyncTime is used in the Python Module 3-WSAG.SyncTime

The Property WsyncTime is used in the Python Module 3-WSAG.SyncTime

The Property ID is used in the Python Module 3-WSAG.ClientSet

The Property Ping is used in the Python Module 3-WSAG.ClientSet and 3-WSAG.SyncTime

The Property PingCheck is used in the Python Module 3-WSAG.SyncTime

The players in Action:

Spawn as Explorer or Your Controlled Client :

Spawn as external Client:

What Information will be send over the Network:

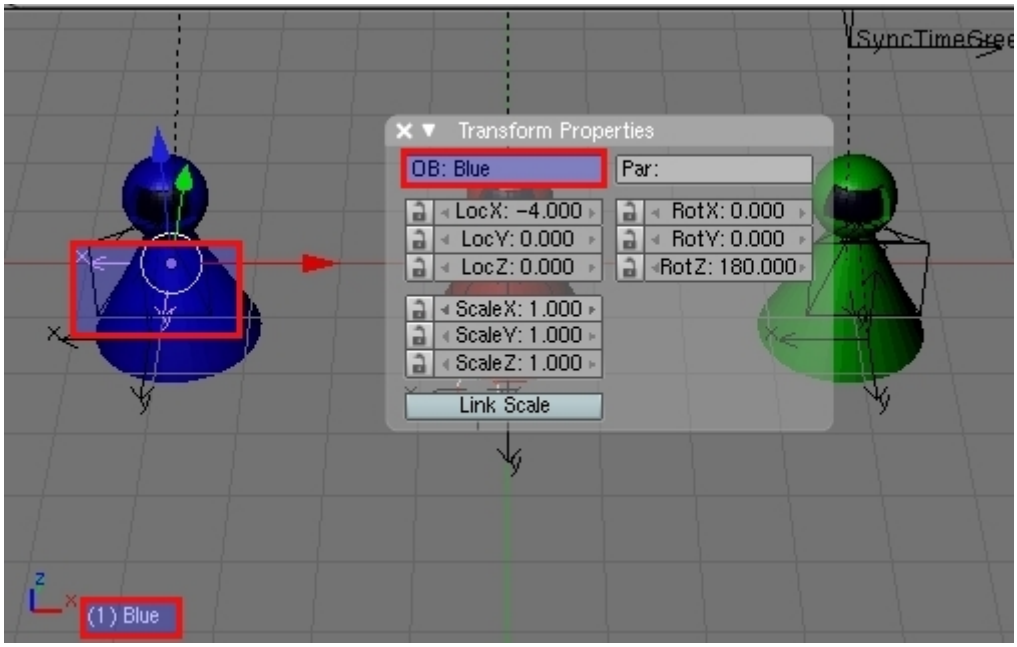
The new WSAG4x Setup makes it very easy for you to define what Information should be send over the network: Simply any property-value of the Player-Element will be send over the network and the receiver will set you your external-clients property whit these values; meaning that your Controlled Client and external-clients propertys will always be in Sync .
So the trick to that you setup the Player-Element so that if it is Spawn as external Client it will use its property's to stay in Sync whit the Player that you control meaning the Player that was Spawn as Your Controlled Client

How to use the Player setup for your Own Game:

Now if you plan an using this setup for your own Game you will most likely what to replace my Player with yours. This is pretty easy as long as you remember some things.
Your Player will need all the Elements described above, for every element is needed for it to work in Multi-player mode. You can have more elements added to your player just make sure you understand how to parent them logical. So that the elements can be clearly separated from deepening if the player is Spawn as a external Client, as Explorer or Your Controlled Client. Se Chapter The player in Action for more information about this.

As a ext steep you have to make sure that the Player is correctly bound in to the setup Menu.

Every Player needs to have a unique Name given to its control Element. See picture below:

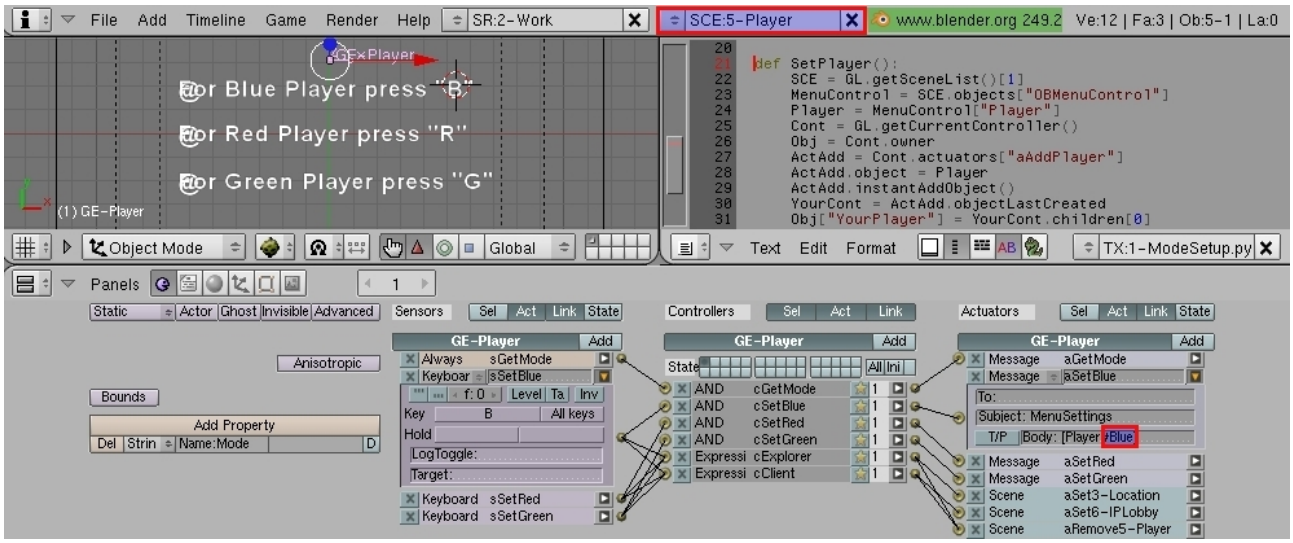


That Unique name will have to be implanted in to the Player selection Menu.
You do that by simply:

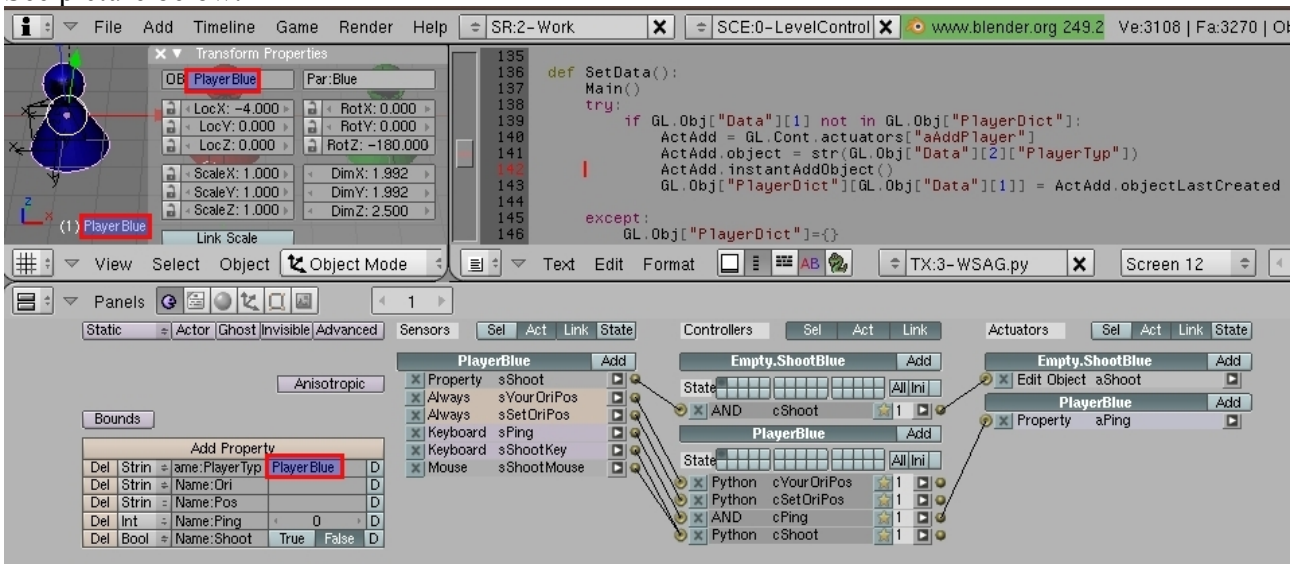
Opening the SCE: 5-Player.
There every player needs a message-actuator that is triggered when the given Player is selected.
That message-actuator has to have a body that is build up like this:
[Player#Blue]

[] = Marks the Beginning and End of the Body
Player = Tels the Script 1-SetMenuProp.py what Property should be set.
# = Marks the Middle of the Body
Blue = Tels the Script 1-SetMenuProp.py what Value should be assigned to the above defined Property

So where in the example above the value Blue is set, you put the unique name of your Player-control-element



As a last step we have to make sure that the Player Type you selected will be transmitted over the network so that it can be created as your external Client on the server and the other client Computers (more information about this can be found in the Chapter Spawn as external Client)
For this to happen you simply need to add the Name of the Player-Element to it's Property: PlayerTyp.
See picture below:



The 1-ModeSetup.py Script

This Script contains the fallowing Modules:

def SetLocation():

def AddMode():

def SetPlayer():

The 1-SetMenuProp.py Script

The 2-MouseMove.py Script

This Script contains the fallowing Modules:

def OnOff():

def MouseY():

def MouseX():

The 2-Move.py Script

The 2-OriPos.py Script

This Script contains the fallowing Modules:

def Main():

def YourOriPosShoot():

def SetOriPosShoot():

def YourOriPos():

def SetOriPos():

The 3-WSAG.py Script

This Script contains the fallowing Modules:

def Main():

def IPLAN():

def Shoot():

def IPWeb():

def Port():

def Receive():

def Send():

def SetIP():

def AppendIP():

def ConnetMe():

def YourConnection():

def SetData():

def SyncTime():

def Disconnect():

def ServerConnect():

def ClientConnect():

def ServerSet():

def ClientSet():